Université de Montréal

Rapport du Projet

Séance été 2023

Xiaoqian 20111352

zoe.wang0505@gmail.com

Travail présenté dans le cadre du cours IFT3150 Projet en informatique

Énoncé du Projet

« Le projet est un projet de développement d'une application mobile par l'organisme MONA, visant la découverte de l'art public, mené par le département d'histoire de l'art et d'études cinématographiques, en coopération avec le département d'informatique et de recherche opérationnelle. L'organisme collabore avec le groupe de recherche en histoire de l'art Art et site de la professeure Suzanne Paquet, directrice du département d'histoire de l'art de l'Université de Montréal. »

Pour ce trimestre, j'ai été recrutée comme développeur iOS. Pendant mon mandat, j'ai travaillé en utilisant le cadre d'applications SwiftUI et le langage Swift avec XCode développés par Apple.

Mandats du Projet

Le mandat pour le projet consiste à publier une nouvelle version d' IOS de MONA pour offrir de nouvelles fonctionnalités:

- Se connecter ou se déconnecter
- Enregistrer un nouveau compte de MONA
- Mettre à jour une utilisateur existant et des données.
- Mettre à jour des données systématiquement dans l'application mobile en utilisant les nouvelles APIs lastUpdatedArtworks et lastUpdatedPlaces, lastUpdatedHeritages.

L'objectif est d'améliorer d'expérience d'utilisateur afin de:

- Retrouver des collections quand l'utilisateur se connecte avec un autre équipement mobile.
- Réduire le temps de téléchargement et la taille des données, quand il y a des données mises à jour sur serveur.
- Prendre en charge tous les types d'appareils IOS avec des tailles d' écran variées, pour les versions IOS 14 et suivantes.

- Mon rôle et ma motivation

J’ai une longue expérience comme développeuse, mais surtout sur des plateformes Windows, avec les langages C++ et C pour des applications desktop, j’ai aussi participé à développer une plateforme de HTML5, mais avec Java et JavaScript.

Ce projet a été la première fois que j'ai eu la chance de travailler dans un environnement d’IOS, et de me familiariser avec le processus de développement de swift 5, car je voulais une expérience sur cette plateforme depuis longtemps.

Travail effectué

- la méthode AGILE

Nous avions un rendez-vous toutes les semaines le lundi après-midi, pour faire un point sur l’avancement de chacun, les problèmes rencontrés et les éventuelles solutions, et partager les ressources nécessaires pour travailler en équipe.Nous avions une réunion sur deux à distance, l’autre au bureau C-8130. Nous avions aussi des groupes de technique sur Élément. On a aussi rencontré une fois l’équipe de désign pour améliorer l’interface d’utilisateur.

- le développement

J’ai passé une bonne partie du début de session à installer l’environnement de travail. Xcode est l’environnement principal. Puis j'ai commencé à apprendre le langage swift 5, à comprendre la manière pour déboguer des fonctions, puis la grammaire et certains des mots clés spécialement conçus pour l’IOS.

Après avoir rencontré Yan, le développeur d’ IOS, qui était en train de résoudre les problèmes causés par la dernière mise à jour d’ application. J’ai eu besoin de comprendre des problèmes pour tester et pour corriger des fonctions.

<u>Introduire de nouvelles APIs</u>

J’ai aussi pris l'initiative d'introduire de nouvelles APIs dans IOS: lastUpdatedArtworks , lastUpdatedPlaces, lastUpdatedHeritages. L’idée est que les données du système vont être de plus en plus grandes, il ne devrait pas être nécessaire de télécharger des données à chaque fois.Lorsqu’il y a des modifications ou de nouvelles données, il est toujours nécessaire de mettre à jour l’application.

Pour ce faire, j'ai eu besoin de comprendre ces situations:

- De quelle manière des données sont sauvegardées dans l'application ? Fichier de texte ou base de données locale?
- De quelle façon des lastUpdated APIs pour des mises à jour ? Avec seulement des propriétés changées, ou, avec une information complète, même si une partie de ces propriétés sont changées.
- Le format de la date sauvegardée dans l'application localement pour IOS.
- J'ai aussi constaté la forme de la date qui généralement est préférable sous la forme d'UTC, car c'est un standard pour résoudre des différences de fuseau horaire. Mais selon des documents d'API, le format de date est yyyy-mm-dd, n'est pas précisé suffisamment.

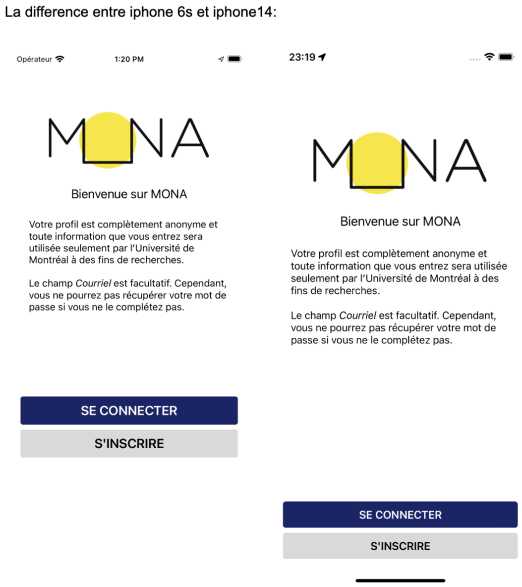
Après avoir lu des documents de serveur et les modèles de structure dans swift, j'ai compris que remplacer des données par ID dans le fichier de JSON est la solution pour les mettre à jour. J'ai choisi de sauvegarder la date en NSDate pour chaque API dans UserDefault. Ces valeurs de la date ne changeront pas si le serveur répond avec erreur.

Corriger une nouvelle interface usager:

Le test est long. J'ai pris une semaine pour tester, noter tous les problèmes que j'ai trouvés, une autre semaine pour les corriger, et recommencer. J'ai testé pour fonctionnalités comme suit:

- Page de bienvenue
  C'est pour des utilisateurs qui ne sont pas connectés à un compte, et aussi après la nouvelle installation.
- Page de mise à jour
  Cette page est pour les utilisateurs de l'ancienne version. Avec l'ancienne version d'IOS, on ne peut pas créer un compte dans le serveur, mais seulement garder des collections localement. Il faut récupérer les données soumises au serveur après qu'ils ont créé un compte.
- Page de registre
  C'était similaire avec la page de mise à jour, mais il est pour les nouveaux utilisateurs.
- Page téléchargée des données
  Il faut prendre du temps pour télécharger des données personnelles et des données systématiquement, après la connexion.
  S'il y a des erreurs à télécharger, l'utilisateur peut le retenter pour une autre fois.
- Fonction de se déconnecter

Cette fonctionnalité est implémentée dans la page de paramètres. Quand l'utilisateur décide de se déconnecter, il faut enlever ses données personnelles et puis afficher la page de bienvenue.



Comme développeuse, au début j'ai pensé que tester était facile. Je me suis concentré sur la logique, et j'avais pensé avoir testé tous les cas pour toutes les situations. Quand j'ai mis l'application sur Testflight pour testeurs internes pour la première fois, j'ai cru que c'était parfait. Mais ensuite, j'ai reçu beaucoup de rapports pour les bogues de l'interface utilisateur, et Lena m'a recommandé de vérifier soigneusement avec le document Figma. Je l'ai suivie et puis j'ai trouvé que c'est vraiment différent entre l'implémentation de UI et le document de la désigner: la police de caractères, les tailles d'image, les styles de boutons, les textes sur les pages et des boutons… Franchement, comme une développeuse depuis longtemps, je n'avais jamais regardé attentivement une interface utilisateur comme celle-là. Et Prof. Guy Lapalme m'a aidé à tester aussi sur le plus petit écran, iPhone 6.

J'ai fait le bilan pour tous les problèmes que j'ai corrigés:

1. J'ai défini des constantes pour des tailles, des couleurs, des rayons de coin,pour rendre tous les composants dans le même style. Donc en changeant seulement les constantes, nous pouvons changer tous les composants.
2. Mettre des positions et des tailles en calculant en fonction de l'écran d'appareil. Donc nous avons des valeurs calculées dynamiquement pour tous les appareils différents.
3. Les réponses d'API ont parfois le même code d'erreur, mais pour des raisons différentes. La solution originale cherchait les phases en anglais correspondant à la phase en français. Cela est un risque pour lequel si l'on modifie les messages sur serveur, cela cause des bogues pour en n'affichant pas des messages valides. J'ai utilisé des codes d'erreurs et j'ai défini des messages pour un ensemble d' erreurs. Par exemple,avec le code 422 pour que le nom d'utilisateur n'existe pas, ou que mauvais mot de passe, un message de <le nom d'utilisateur n'existe pas ou le mot de passe n'est pas valide> être affiché, à place d'un message général <erreur du serveur>.

4. Parfois, j'ai rencontré le problème de la mise à jour incorrecte des données sur le serveur. Ce qui m'a donné une erreur inattendue dans les mobiles. J'ai aidé Jianxin pour trouver les problèmes, et par cela, j'ai mieux compris des structures de données de MONA.

Page de mot de passe oublié:

Actuellement, c'est une partie de fonctionnalité pour la page de connexion. Mais, l'API n'avait pas été implémenté à ce moment-là. Nous avions prévu de masquer l'option pour la nouvelle version. Mais après une rencontre avec l'équipe, JianXin a préparé l'API dans une semaine. Donc pour tester cet API, j'ai implémenté la page de mot de passe oublié, donc nous avons pu publier une solution complète.

Un problème noté pour IOS token:

Nous avons vu le problème quand nous avons enlevé l'application de MONA sans nous déconnecter, puis le télécharger et ouvrir, nous ne pouvons pas l'utiliser, parce que l'application considère qu'il s'est connecté, mais aucune donnée n'existe. Après avoir cherché sur internet, j'ai trouvé que c'était à cause du token d'utilisateur sauvegardé dans le cache local. Nous avions fait en sorte de la logique pour le nettoyer quand l'utilisateur se déconnecte de l'application, mais pas lorsque l'utilisateur enlève l'application. Et le token ancien est encore valide jusqu'à ce que l'utilisateur se reconnecte avec un autre appareil.Selon Apple développer, la solution est de sauvegarder des tokens à serveur,à la place du cache local, et plus avec le serveur vérifie le token à accéder aux données, et génère un nouveau token pour la sécurité. Mais c'est un autre niveau de solution sécurité que nous pourrons avoir dans le futur.

Conclusion

En général, mon expérience dans notre équipe Mona a été très intéressante, car j'étais développeur et testeur en même temps pour la première fois, et nous avons travaillé comme une équipe: j'ai travaillé avec Jianxin pour corriger des erreurs de données pour serveur, et j'ai travaillé avec Kim pour construire la version IOS par Ionic et Vue.js.

C'est la première fois que j'ai travaillé avec une interface usager (UI) et avec la logique en même temps, et j'ai compris très sérieusement l'importance de UI pour des utilisateurs, pendant les itérations de tests internes. Il n'a pas été facile de changer mon point de vue de développeur à utilisateur, mais, car nous travaillions dans une petite équipe, j'ai changé mon rôle pour chaque semaine pendant les itérations, ce fut une expérience très importante.

Durant ces quatre mois, j'ai terminé le processus d'ajout de nouvelles fonctionnalités, tester, corriger des bogues, et enfin, publier une nouvelle version de MONA sur Apple App Store. Je suis très content de cela,surtout au moment où la nouvelle version est prête pour télécharger dans L'App Store.

J'espère que c'est la dernière version de swift pour MONA, et l'application de Ionic et Vue.js va probablement la remplacer très prochainement. J'espère aussi que ce projet de devenir de plus en plus populaire après ce bel effort de Lena et de toute l'équipe.